SEVENSTAX SMTP

User Manual

Revision No.:	5.03
State:	Draft
Author:	SEVENSTAX GmbH

Initial version:	21/08/02
Last change:	30/10/09
Last Review:	
Publication:	Public
File name:	SEVENSTAX_SMTP_UserManual_V503

Copyright (c) 2002-2009 by SEVENSTAX GmbH
This document is an intellectual property of SEVENSTAX GmbH. Unauthorized copying and distribution is prohibited.

Table of Contents

1 Preface....4
2 Protocols Overview....5
3 Integration into User Application....6
4 SMTP Control Module....7
4.1 Overview....7
4.2 Protocol API....8
4.2.1 stxSMTPControl_Init (fNotifyPara)....8
4.2.2 stxSMTPControl_Tick ( )....8
4.2.3 stxSMTPControl_Configure (szServerName, uServerPort, szLogin, szPassword)....8
4.2.4 stxSMTPControl_SendMail (hConnNetwork, szMailAddress, szToMailAddresses, szCCMailAddresses, szBCCMailAddresses, szSubject, szBody, rgsAttachments, uCntAttachments)....9
4.2.5 stxSMTPControl_IsBusy ( )....10
4.3 Notify Codes....11
4.3.1 NC_SMTPCNTL_STATE_INFO (actual_state , -)....11
4.3.2 NC_SMTPCNTL_OP_READY (- , -)....11
4.3.1 NC_SMTPCNTL_OP_FAILED (uSMTPControl_Err , -)....11
4.4 Adjustable Parameters....11
5 IMF Generator Module....12
5.1 Overview....12
5.2 Protocol API....13
5.2.1 stxIMFGen_InitData ( )....13
5.2.2 stxIMFGen_SetSenderAddress (szSenderMailAddressPara)....13
5.2.3 stxIMFGen_SetRcptToAddresses (szRcptMailAddressesPara)....13
5.2.4 stxIMFGen_SetSubject (szSubjectPara)....14
5.2.5 stxIMFGen_SetMailDate (pdDateSendPara)....14
5.2.6 stxIMFGen_SetMailTime (pdTimeSendPara)....14
5.2.7 stxIMFGen_SetMailText (szMailTextPara)....15
5.2.8 stxIMFGen_SetMailAttachments (rgszSendAttContentsPara, rgszSendAttFilenamesPara, uCntAttachmentsPara)....15
5.2.9 stxIMFGen_GetEncodedMail (uMode, szTarget, puSizeTarget)....16
6 SMTP Module....17

6.1 Overview....17
6.2 Protocol API....18
6.2.1 stxSMTP_Init (fctNotifyPara)....18
6.2.2 stxSMTP_RegHndlFct_SendData (fctSendDataPara)....18
6.2.3 stxSMTP_RegHndlFct_GetMailPkt (fctGetMailPara)....18
6.2.4 stxSMTP_Reset ( )....19
6.2.5 stxSMTP_Tick ( )....19
6.2.6 stxSMTP_SetMailData (szMailRawDataPara)....19
6.2.7 stxSMTP_SendMail (szSMTPServerUsernamePara, szSMTPServerPasswordPara, szSenderMailAddressPara, szRcptPara)....20
6.2.8 stxSMTP_ReadyToSend ( )....21
6.2.9 stxSMTP_Parse (pBuffer, uAnzahl)....21
6.3 Notify Codes....22
6.3.1 NC_SMTP_READY (- , -)....22
6.3.2 NC_SMTP_ERRCODE (usSMTP_ErrCode , eSMTP_State)....22
6.3.3 NC_SMTP_NO_ANSWER (- , -)....22
6.3.4 NC_SMTP_WRONG_STATE (- , eSMTP_State)....23
6.3.5 NC_SMTP_SEND_DATA_START (- , -)....23
7 Debugging Support....24
8 Restrictions....25
9 Change History....26

1 Preface

SEVENSTAX SMTP is an implementation of the SMTP protocol. SMTP stands for "Simple Mail Transfer Protocol" and is the world wide standard protocol for sending e-mails. The implementation is designed for use in microcontrollers but can also be used on any other platform.

The main tasks of SEVENSTAX SMTP are to build and send SMTP commands and mail data to the SMTP server and to interpret the responses from the SMTP server. SEVENSTAX SMTP is designed to work on top of SEVENSTAX TCP/IP. It also includes a control module that handles resolving the server name and manages the TCP connection.

Furthermore, a MIME encoding module is part of SEVENSTAX SMTP. This may be use to realise mail encoding (with respect to the Internet message format and the MIME standard) and delivering the mail in packets. In this case a corresponding handler function must be registered. Please note that MIME does not only support the usage of attachments but also of special characters. SMTP supports only 7 bit ASCII characters. There is an enhanced version of SMTP which is capable to process 8 bit ASCII characters, too. Although this enhanced SMTP runs on many current SMTP servers there is no guarantee of an 8 bit ASCII support.

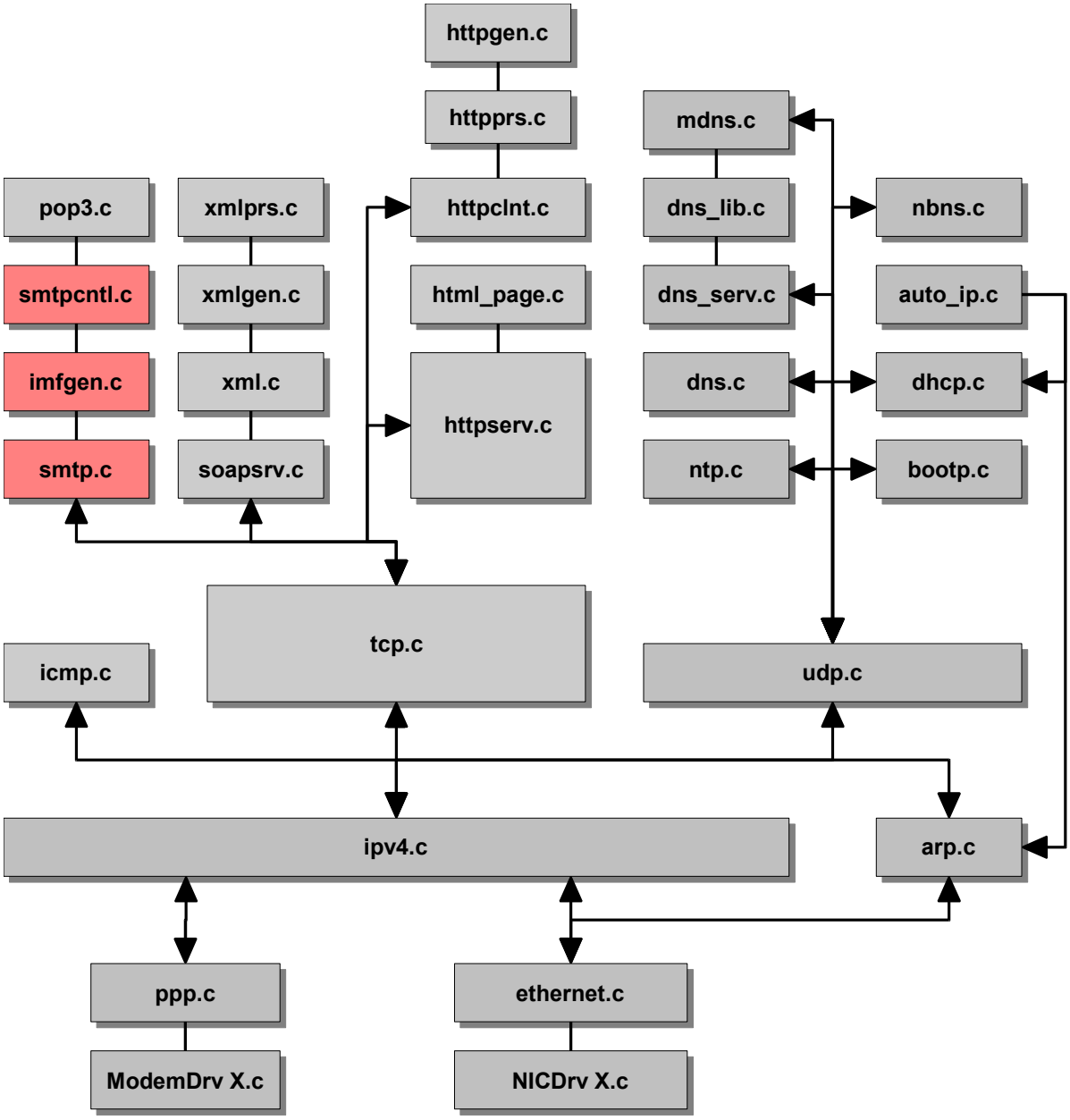
If you do not want to send an encoded mail, you can use SEVENSTAX SMTP to send uncoded strings or a sequence of characters directly. Most SMTP servers will add corresponding header fields and send the mail correctly although the mail could have 8 bit coded characters.

SEVENSTAX SMTP provides sending mails over SMTP servers that accept the authentication method "plain" or that don't need any authentication.

The application should include a notify handler function to be called from SEVENSTAX SMTP to request data (optional, if not passed to the send mail function directly), on successful termination, or on error.

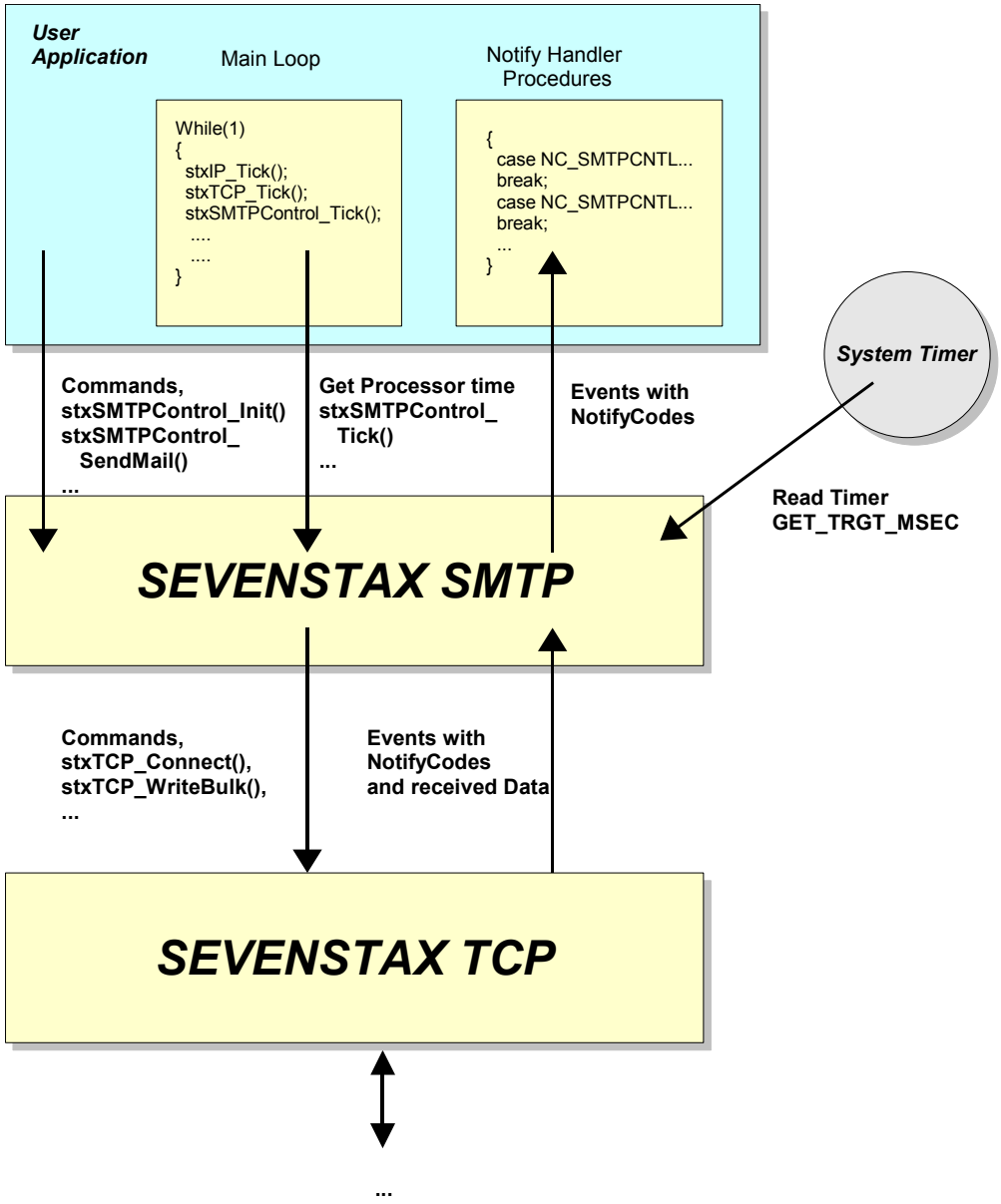
2 Protocols Overview

The following graphic shows the integration of SEVENSTAX SMTP (red marked) inside the other SEVENSTAX internet libraries, available as separate products.



3 Integration into User Application

SEVENSTAX SMTP is designed to work in conjunction with SEVENSTAX TCP/IP. The picture below gives an overview of the interaction of the user application with SEVENSTAX products.



For further details on configuration options of the features.h file and notify handler basics, please refer to the SEVENSTAX TCP/IP User Manual.

4 SMTP Control Module

4.1 Overview

An easy to use control module is included in SEVENSTAX SMTP. Using this module will take complexity out of the user application, since the control module already handles resolving the SMTP server name into the corresponding IP address and manages the TCP connection by calls to lower layers in the stack.

SMTP can be enabled by setting *SMTP_SUPPORTED* to 1 in *features.h* (see also the SEVENSTAX TCP/IP User Manual).

The SMTP protocol works based on the name service module and on top of TCP, so *NAMESERV_SUPPORTED*, either *DNS_SUPPORTED* or *DNS_SRV_SUPPORTED*, *UDP_SUPPORTED*, and *TCP_SUPPORTED* must also be set to 1. It is also recommended to set *IMFGEN_SUPPORTED* to 1. This will enable the proper generation using the Internet Message Format.

SMTP uses TCP on port 25 for transmission.

To send e-mail via SMTP server with SEVENSTAX SMTP you should proceed in the following way:

1. Set *SMTP_SUPPORTED* to 1 in *features.h* (assuming all other required modules are also enabled)
2. Call *stxSMTPControl_Init()* once to set up internal state of SEVENSTAX SMTP.
3. Call *stxSMTPControl_Configure*() to set the SMTP server name, port, login, etc.
4. Call *stxSMTPControl_SendMail()* to start sending an e-mail.
5. In your main loop or operating system task call *stxSMTPControl_Tick()* repeatedly in order to keep alive and finish sending the e-mail.
6. After a while, the handler procedure may receive *NC_SMTPCNTL_STATE_INFO* events that indicate the progress of the emailing.
7. Finally, the handler procedure will receive either an *NC_SMTPCNTL_OP_READY* or *NC_SMTPCNTL_OP_FAILED* event.
8. On *NC_SMTPCNTL_OP_READY* the e-mail transmission was successfully completed.
9. On *NC_SMTPCNTL_OP_FAILED* the e-mail operation failed. In this case an error code is also passed to the notify handler to indicate the cause of failure.

4.2 Protocol API

4.2.1 stxSMTPControl_Init (fNotifyPara)

Description

Sets up some internal variables and registers the application callback function. Must be called once before using any other SMTP control function.

Parameter

PROTOCOL_NOTIFY_HANDLER *fNotifyPara*
Application handler function to receive notify codes from SEVENSTAX SMTP module. The handler function will be called as sending mail progresses and after operation is finished successfully or due to failure.

Return Value

-

4.2.2 stxSMTPControl_Tick ( )

Description

This function keeps SMTP alive, assuring multiple retries until a timeout occurs. It should repeatedly be called by the application, like all the other sevenstax “Tick” functions, whether or not there is an ongoing SMTP operation.

Parameter

-

Return Value

-

Comment

This function will call the application notify handler with the intermediate state of progress (if applicable) and the result of the termination of the operation.

4.2.3 stxSMTPControl_Configure (szServerName, uServerPort, szLogin, szPassword)

Description

The SMTP account is configured by calling this function. Sets some internal variables and returns immediately.

Parameter

CHAR_stx FPTR_stx *szServerName*
Server name of SMTP server.

UINT16_stx *uServerPort*
Server TCP port (usually 25).

CHAR_stx FPTR_stx *szLogin*
Login name for authentication.

CHAR_stx FPTR_stx *szPassword*
Login password for authentication.

Return Value

-

4.2.4 stxSMTPControl_SendMail (hConnNetwork, szMailAddress, szToMailAddresses, szCCMailAddresses, szBCCMailAddresses, szSubject, szBody, rgsAttachments, uCntAttachments)

Description

Starts sending a mail over an SMTP server. Sets some internal variables and returns immediately. The actual operations will be done during during the periodic calls of *stxSMTPControl_Tick()*.The application defined handler procedure will receive the notify codes as described below

Parameter

CONN_HNDL_TYPE *hConnNetwork* (only multi device version)
Connection handle of underlying network layer over which the e-mail should be sent.

CHAR_stx FPTR_stx *szMailAddress*
Own e-mail address ("From" field) - short or extended format.

CHAR_stx FPTR_stx *szToMailAddresses*
List of recipients ("To" field) - comma separated; each short or extended format.

CHAR_stx FPTR_stx *szCCMailAddresses*
List of recipients ("CC" field) - comma separated; each short or extended format.

CHAR_stx FPTR_stx *szBCCMailAddresses*
List of recipients ("BCC" field) - comma separated; each short or extended format.

CHAR_stx FPTR_stx *szSubject*
The e-mail subject line.

CHAR_stx FPTR_stx *szBody*
The message body.

SMTPCNTL_ATT FPTR_stx *rgsAttachments*
List of attachments. Each element contains a pointer to a filename and a pointer to the contents.

UINT8_stx *uCntAttachments*
Number of attachments. The maximum number of attachments is predefined (see also section 4.4)

Return Value

BOOL_stx
TRUE_stx, if the send request was accepted
FALSE_stx, if the send request was not accepted (SMTP Control module busy)

Comment

Please note: Delivered data blocks must not be changed until send task has finished (received NC_SMTPCNTL_OP_READY or NC_SMTPCNTL_OP_FAILED)

The following formats may be used for e-mail addresses:
short format: "info@sevenstax.de"
extended format: "Sevenstax Info <info@sevenstax.de>"

4.2.5 stxSMTPControl_IsBusy ( )

Description

Returns, if the SMTP control module is busy.

Parameter

-

Return Value

BOOL_stx

TRUE_stx,	there is an active emailing operation
FALSE_stx,	no emailing operation is currently active

4.3 Notify Codes

4.3.1 NC_SMTPCNTL_STATE_INFO (actual_state , -)

This event indicates an information on the progress of the emailing operation.

Parameter 1 is used to deliver reason of failure:
SMTPCNTL_STATE_SERVERNAME_RESOLVED
SMTPCNTL_STATE_SERVER_CONNECTED

Parameter 2 is not used and always UINT32_stx_MAX.

ReturnValue not used.

4.3.2 NC_SMTPCNTL_OP_READY (- , -)

This event indicates a successful emailing operation.

Parameter 1 is not used and always UINT32_stx_MAX.

Parameter 2 is not used and always UINT32_stx_MAX.

ReturnValue not used.

4.3.1 NC_SMTPCNTL_OP_FAILED (uSMTPControl_Err , -)

The e-mail operation failed due to the error as indicated by *Parameter 1*.

Parameter 1 is used to deliver reason of failure:
SMTPCNTL_ERROR_SERVERNAME_NOT_RESOLVED
SMTPCNTL_ERROR_TCP_CONN_REFUSED
SMTPCNTL_ERROR_TCP_CONN_FAILED
SMTPCNTL_ERROR_SMTP_SEND_REFUSED
SMTPCNTL_ERROR_SMTP_SEND_FAILED

Parameter 2 is not used and always UINT32_stx_MAX.

ReturnValue not used.

4.4 Adjustable Parameters

SMTPCNTL_TIMEOUT_NAME_RESOLVE **default: 60**
valid: 1..65535

With this macro you can define the maximum time in seconds a name service query may take. The allowed range is given above, but it is not recommended to allow for more time then a couple of minutes.

SMTPCNTL_NUM_ATTACHMENTS_MAX **default: 5**
valid: 1..

With this macro you can define the maximum number attachments. This is used to internally allocate memory for pointer lists, so increasing this number will increase the memory usage of the module.

5 IMF Generator Module

5.1 Overview

This module is an encoder for text messages that are sent between computer users, within the framework of "electronic mail" messages. IMF stands for "Internet Message Format" and is the world wide standard format for electronic mails. To ensure a delivery via all existing mail servers in the world the mail is encoded so that all characters are in 7 bit format. This encoding is defined in the MIME (Multipurpose Internet Mail Extensions) standard which also defines mechanism to integrate attachments into the mail. The implementation is designed for use in microcontrollers but can also be used on any other platform.

The main task of the IMF generator module is to provide encoded mail packets (of user defined size). These mail packets can for example be concatenated to a whole mail or can be sent directly via SMTP to a mail server (streaming). Neither the sending of the mail nor the connection to a mail server is part of this module. It serves only as a source of encoded mail. To send an encoded mail over SMTP you can use the SMTP control module which will handle the communication with a server.

Before the IMF module can provide packets of the encoded mail the user has to deliver all relevant mail properties to this module. This are e.g. the sender, the recipients, the subject and the mail text. To do this, the module provides special set functions. In case, the SMTP control module is used, this task is performed internally and there is no need to call any functions of the IMF generator module directly.

When using the SMTP control module (see chapter 4), *no* function of this module should be called directly by the user application! The functions defined in the API below are called internally by the SMTP control module, whenever required.

5.2 Protocol API

5.2.1 stxIMFGen_InitData ( )

Description

This function resets all mail properties. These are the sender mail address, the list of recipient mail addresses, the subject, date and time, the mail text, the list of attachment contents, file names, and the count of attachments.
To set these properties use the appropriate set function.

Parameter

-

Return Value

-

5.2.2 stxIMFGen_SetSenderAddress (szSenderMailAddressPara)

Description

This function sets the mail address of the sender.

Parameter

STRING_stx *szSenderMailAddressPara*
You can either use just the mail address (e.g. "`tom@domain.com`") or the enhanced mail address format with a nickname including umlauts (e.g. "`Rüdiger Mustermann <ruedigerm@sevenstax.de>`").

Return Value

-

Comment

Only the pointer to the string is stored – so the data must not be changed during the mail encoding process.

5.2.3 stxIMFGen_SetRcptToAddresses (szRcptMailAddressesPara)

Description

This function sets the list of mail addresses of the mail recipients.

Parameter

STRING_stx *szRcptMailAddressesPara*
List of recipients. The recipients are separated by commas. For every recipient you can either use just the mail address or the enhanced mail address format.
Example: "`Tim <tim@domain.com>, lucy@domain.com, Rüdiger Mustermann <ruedigerm@sevenstax.de>`".

Return Value

-

Comment

Only the pointer to the string is stored – so the data must not be changed during the mail encoding process.

5.2.4 stxIMFGen_SetSubject (szSubjectPara)

Description

This function sets the subject of the mail.

Parameter

STRING_stx *szSubjectPara*
The mail subject. The string can include umlauts, e.g. "`A hint to handle umlauts like Ö`".

Return Value

-

Comment

Only the pointer to the string is stored – so the data must not be changed during the mail encoding process.

5.2.5 stxIMFGen_SetMailDate (pdDateSendPara)

Description

This function sets the date that is to be used in the mail.

Parameter

DATE_TYPE FPTR_stx *pdDateSendPara*
The date of the e-mail. The parameter is a pointer to a structure which contains the members year, month and day. The values are used as is, e.g. {3, 12, 24} stands for December 24th, 2003.

Return Value

-

Comment

If not both, date and time, are set, the encoding will not generate the “Date:” field. The mail user agent should set the date. However, if the date field is not provided, almost all mail servers will insert this.

5.2.6 stxIMFGen_SetMailTime (pdTimeSendPara)

Description

This function sets the local time that is to be used in the mail.

Parameter

TIME_TYPE FPTR_stx *pdTimeSendPara*
The date of the e-mail. The parameter is a pointer to a structure which contains the members hour, minute, second, difference of hours to Greenwich time (GMT), difference of minutes to GMT and a flag if we are later than GMT or not.
The values are used as is, e.g. {23, 30, 42, 2, 0, TRUE} for local time of 23:30:42h and 2:00 hours later than GMT.

Return Value

-

Comment

If not both, date and time, are set, the encoding will not generate the “Date:” field. The mail user agent should set the date. However, if the date field is not provided, almost all mail servers will insert this.

5.2.7 stxIMFGen_SetMailText (szMailTextPara)

Description

This function sets the text of the mail body.

Parameter

STRING_stx *szMailTextPara*
The mail body. The string can include umlauts, e.g. "`A hint to handle umlauts like Ö is: use quoted printable encoding`".

Return Value

-

Comment

Only the pointer to the string is stored – so the data must not be changed during the mail encoding process.

5.2.8 stxIMFGen_SetMailAttachments (rgszSendAttContentsPara, rgszSendAttFilenamesPara, uCntAttachmentsPara)

Description

This function sets the text attachments of the mail. Here is an Example: `stxIMFGen_SetMailAttachments( {"attachment 1 text", "attachment 2 text"}, {"Att1.txt", NULL}, 2 )` means two attachments, where the first attachment has a file name and the second one has no file name.

Parameter

STRING_stx FPTR_stx *rgszSendAttContentsPara*
A pointer to an array with the text contents of the attachments.

STRING_stx FPTR_stx *rgszSendAttFilenamesPara*
A pointer to an array which contains the file names that are used by the mail client of the recipient to show and save the attachments. If no file name has to be used, the corresponding entry must be NULL_stx.

UINT8_stx *uCntAttachmentsPara*
The total number of attachments.

Return Value

-

Comment

Only the pointers to the strings are stored – so the data must not be changed during the mail encoding process.

5.2.9 stxIMFGen_GetEncodedMail (uMode, szTarget, puSizeTarget)

Description

With this function, the user can request the packets of the encoded mail one after another.

Parameter

UINT8_stx *uMode*
If this parameter is equal to zero, the first encoded mail packet is returned. For all other values, the next packet is returned.

STRING_stx *szTarget*
A pointer to the buffer into which the returned packet will be copied.

UINT16_stx FPTR_stx *puSizeTarget*
A pointer to the size of the delivered buffer.

Return Value

-

Comment

The function always fills the buffer completely, except for the last packet. The content of the third parameter is overridden with the count of chars that are written into the buffer. So, if this value is less than the originally delivered value, the last packet was returned by the IMF generator module.

6 SMTP Module

6.1 Overview

This module contains an implementation of the SMTP protocol: it is used to build and send SMTP commands and mail data to the SMTP server and to interpret the responses from the server.

Not part of this module is to establish a TCP connection to the SMTP server and to send data over TCP. Please use the SMTP control module to accomplish these tasks.

Also the MIME encoding of the mail is not part of this module. Please use the IMF generator module to accomplish this task.

When using the SMTP control module (see chapter 4), *no* function of this module should be called directly by the user application! The functions defined in the API below are called internally by the SMTP control module, whenever required.

6.2 Protocol API

6.2.1 stxSMTP_Init (fctNotifyPara)

Description

Sets up internal variables and registers the notify callback function. Must be called once before using any other SMTP function.

Parameter

PROTOCOL_NOTIFY_HANDLER *fctNotifyPara*
Handler function to receive notify codes from SMTP module. The handler function will be called as sending mail progresses and after operation is finished successfully or due to failure.

Return Value

-

6.2.2 stxSMTP_RegHndlFct_SendData (fctSendDataPara)

Description

Registers a send data handler function.

Parameter

SMTP_SENDDATA_FCTTYPE *fctSendDataPara*
Pointer to send data function.

Return Value

-

Comment

The SMTP module calls this handler function when it wants to send data to the SMTP server. Normally you will use TCP as transport layer. As many TCP write functions (like the one of SEVENSTAX TCP/IP) require additional parameters (e.g. a connection id) they can not be registered directly to SMTP. Instead an application-specific write function should be used that knows the connection id and that satisfies the SMTP write function requirements.

6.2.3 stxSMTP_RegHndlFct_GetMailPkt (fctGetMailPara)

Description

Registers a get mail data handler function..

Parameter

SMTP_GETMAIL_FCTTYPE *fctGetMailPara*
Pointer to get mail function.

Return Value

-

Comment

The SMTP module calls this handler function when it wants to send the next (where applicable encoded) mail packet, so encoding can be processed in a streaming manner. The function stxIMFGen_GetEncodedMail() of the IMF generator module could be used as the get mail function to deliver e-mail data with standard MIME (Multipurpose Internet Mail Extensions) encoding.

6.2.4 stxSMTP_Reset ( )

Description

Resets the internal state SMTP module to “idle”.

Parameter

-

Return Value

-

Comment

This function must be called every time when a (TCP) connection to the SMTP server has been established. This is necessary because the SMTP module does not handle the server connection on the transport layer (TCP). If a new (TCP) connection has been established (and thus this reset routine was called) the SMTP module knows that it has to connect to the SMTP server on SMTP level (waiting for a greeting message, sending "HELO" command etc.).

Calling this function is also performed automatically whenever required by the SMTP control module.

6.2.5 stxSMTP_Tick ( )

Description

This function keeps the SMTP sub-module alive. It should repeatedly be called, whether or not there is an ongoing SMTP operation. In case the SMTP control module is used (recommended!), this tick function will internally be called from the tick function of the control module.

Parameter

-

Return Value

-

6.2.6 stxSMTP_SetMailData (szMailRawDataPara)

Description

If you do not want to use a mail encoder but only want to send a raw constant string directly via SMTP, you can specify this string in this function.

Parameter

STRING_stx *szMailRawDataPara*
The data to send.

Return Value

-

Comment

This string must remain unchanged until the registered notify handler function is called with a notify code which states that the SMTP process is ready or has failed. This is because the SMTP module does not copy the delivered string but accesses it directly to save memory.

6.2.7 stxSMTP_SendMail (szSMTPServerUsernamePara, szSMTPServerPasswordPara, szSenderMailAddressPara, szRcptPara)

Description

This function is used to send a mail via SMTP.

Parameter

STRING_stx *szSMTPServerUsernamePara*
User name of the SMTP server account. Maximum length: 65535 chars.

STRING_stx *szSMTPServerPasswordPara*
Password of the SMTP server account. Maximum length: 65535 chars.

STRING_stx *szSenderMailAddressPara*
Sender of the mail. This can be just a mail address or the extended format that consists of a name followed by a '<', followed by the mail address and terminated by a '>' (e.g. "Nick <nick@name.xy>"). Maximum length: 255 chars.

STRING_stx *szRcptPara*
List of recipients, separated by commas. Each recipient can be specified by just the mail address or by using the extended format, e.g. "Näck <naeck@name.xy>, nock@name.xy". Maximum length: 255 chars.

Return Value

SMTP_ERR

SMTP_ERR_BUSY,	failed to start another emailing operation – application must wait until notify function is called.
SMTP_ERR_OK,	emailing was started and will run asynchronously.

Comment

User name and password are used for the authentication process where applicable. If you are sure that your SMTP server does not need authentication you do not have to specify these parameters.

All parameter strings **must** remain unchanged until the registered notify handler function was called with a notify code which states that the SMTP process is ready or has failed. This is because the SMTP module does not copy the delivered strings but accesses them directly to save memory.

6.2.8 stxSMTP_ReadyToSend ( )

Description

This function is called from the application (or if used: by the SMTP control module) if it has determined that the current data packet was successfully sent to the server via transport layer (TCP). For the SMTP module this means that it may for example build the next packet and send it.

Parameter

-

Return Value

-

6.2.9 stxSMTP_Parse (pBuffer, uAnzahl)

Description

This function must be called every time when a data packet from the SMTP server was received.

Parameter

STRING_stx *pBuffer*
Pointer to a buffer with the received data.

UINT16_stx *uAnzahl*
Number of Bytes in the packet.

Return Value

-

6.3 Notify Codes

6.3.1 NC_SMTP_READY (- , -)

This event indicates an information on the progress of the emailing operation.

Parameter 1 is not used and always UINT8_stx_MAX.

Parameter 2 is not used and always UINT8_stx_MAX.

ReturnValue not used.

6.3.2 NC_SMTP_ERRCODE (usSMTP_ErrCode , eSMTP_State)

This event indicates a successful emailing operation.

Parameter 1 is used to deliver the SMTP error code.
(see rfc2821, rfc1123, rfc1893, and rfc2034)

Parameter 2 is used to deliver the state of the SMTP module:
SMTP_STATE_IDLE_NOCOM
SMTP_STATE_CONNECT
SMTP_STATE_HELO
SMTP_STATE_AUTH
SMTP_STATE_IDLE_COM
SMTP_STATE_MAIL
SMTP_STATE_RCPT
SMTP_STATE_DATA
SMTP_STATE_SENDING
SMTP_STATE_RSET
SMTP_STATE_VRFY
SMTP_STATE_EXPN
SMTP_STATE_HELP
SMTP_STATE_NOOP
SMTP_STATE_QUIT

ReturnValue not used.

6.3.3 NC_SMTP_NO_ANSWER (- , -)

This event indicates that a timeout occurred while waiting for an answer of the SMTP server.

Parameter 1 is not used and always UINT8_stx_MAX.

Parameter 2 is not used and always UINT8_stx_MAX.

ReturnValue not used.

6.3.4 NC_SMTP_WRONG_STATE (- , eSMTP_State)

This event indicates a problem with the state machine of the SMTP module.

Parameter 1 is not used and always UINT8_stx_MAX.

Parameter 2 is used to deliver the state of the SMTP module.

ReturnValue not used.

6.3.5 NC_SMTP_SEND_DATA_START (- , -)

This event indicates that all recipients passed and that sending data may now begin.

Parameter 1 is not used and always UINT8_stx_MAX.

Parameter 2 is not used and always UINT8_stx_MAX.

ReturnValue not used.

7 Debugging Support

At source level, SEVENSTAX TCP/IP contains a simple debugging support through macros mapped to printf() (DebugOut-Strings).

The main switch to enable DebugOuts for all modules is DEBUG_STX = 1 in *features.h*.

```
/* basic debug settings */
#define DEBUG_STX               1       /* Main switch: sevenstax debugging enabled */
```

The SMTP module has a separate switches to enable debug information at compilation time. This enables a fine control about the debug information you will get. You can control it through the settings in *features.h*:

```
...
/* hl protocols debug */
...
#define SMTPCNTL_DEBUG          1       /* SMTP control module */
#define SMTP_DEBUG              1       /* SMTP module */
#define IMFGEN_DEBUG            1       /* IMF (Internet Message Format) generator */
...
```

For more information about debugging features in the SEVENSTAX protocol stack please refer to the SEVENSTAX TCP/IP & SEVENSTAX Routing User Manual.

8 Restrictions

SEVENSTAX SMTP is focused on systems with very small resources, to give them relevant email functionality. There are some restrictions in usage and capabilities as listed below.

#	*Restriction*	*Details*
1	Number of Attachments	May be configured by user; fixed at compile time
2	Mime Types	Only plain text attachments are supported.
3	Authentication to SMTP server	Authentication method "plain" supported. Also, supported are servers that don't need any authentication
4	E-Mail header fields	User-Agent, From, To, Subject, MIME-Version, Content-Type, Content-Transfer-Encoding. Currently, there is no support for Notifications, Priority, CC and BCC.
5	Date - SMTP control module	Although it is possible to set the date for generated messages, the high level control module does not currently pass date information to the IMF encoder.

9 Change History

Ver.	*Date*	*by*	*Change description*
5.01	21. Oct. 2009	jma	Base version
5.02	29. Oct. 2009	jma	Interface changes (stxSMTPControl_Configure, stxSMTPControl_SendMail)
5.03	30. Oct. 2009	jma	Added IMF generator, and SMTP module API

The information furnished in this document is believed to be accurate and reliable. However, no responsibility is assumed by SEVENSTAX for its use, nor for any infringements of patents or other rights of third parties resulting from its use. No license is granted under any patents or patent rights of SEVENSTAX.

This document is an intellectual property of SEVENSTAX GmbH. Unauthorized copying and distribution is prohibited.

Copyright (c) 2002-2009 by SEVENSTAX GmbH